Charge sharing for MAPS detector pixels

Tony Frawley
FSU

sPHENIX Simulations Meeting
3/14/2017

Why we need charge sharing for MAPS

The existing implementation of the MAPS pixel readout does not include charge sharing.

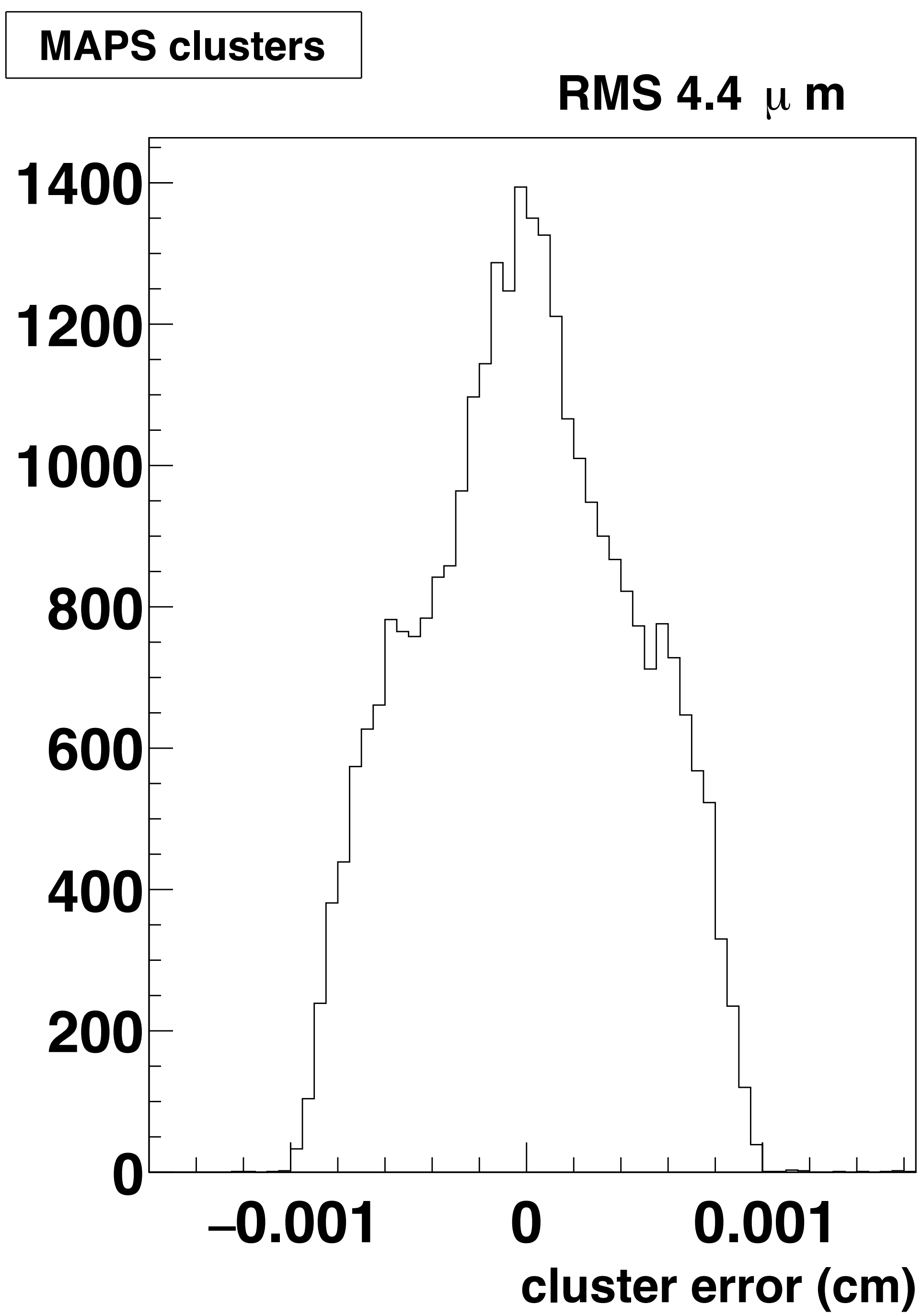
This was allowed for in our previous simulations (by Mike) by reducing the pixel size from 30 microns to 20 microns, so that the effective resolution of 20 microns/$\sqrt{12}$ is about equal to what we would expect with 30 micron pixels and charge sharing. This worked well enough for the cylinder cell geometry.

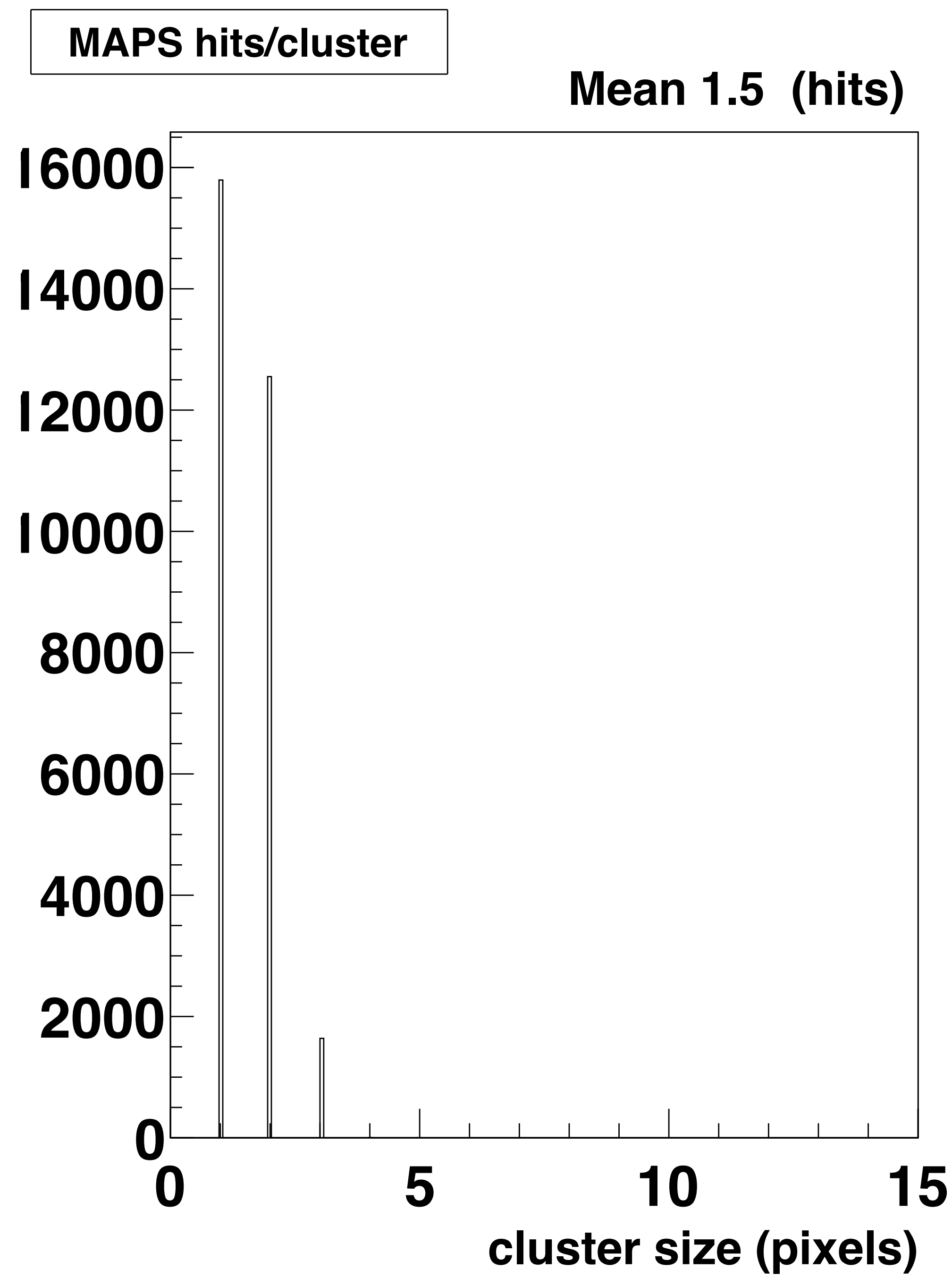
However there is a problem with this when we use the realistic model of the MAPS detector.

The ladders are tilted so, unlike the cylinder cell model, we get a substantial number of tracks that pass through two or more pixels. This allows the clustering to get a much better position resolution for some hits, as shown in the next few slides.

Without charge sharing

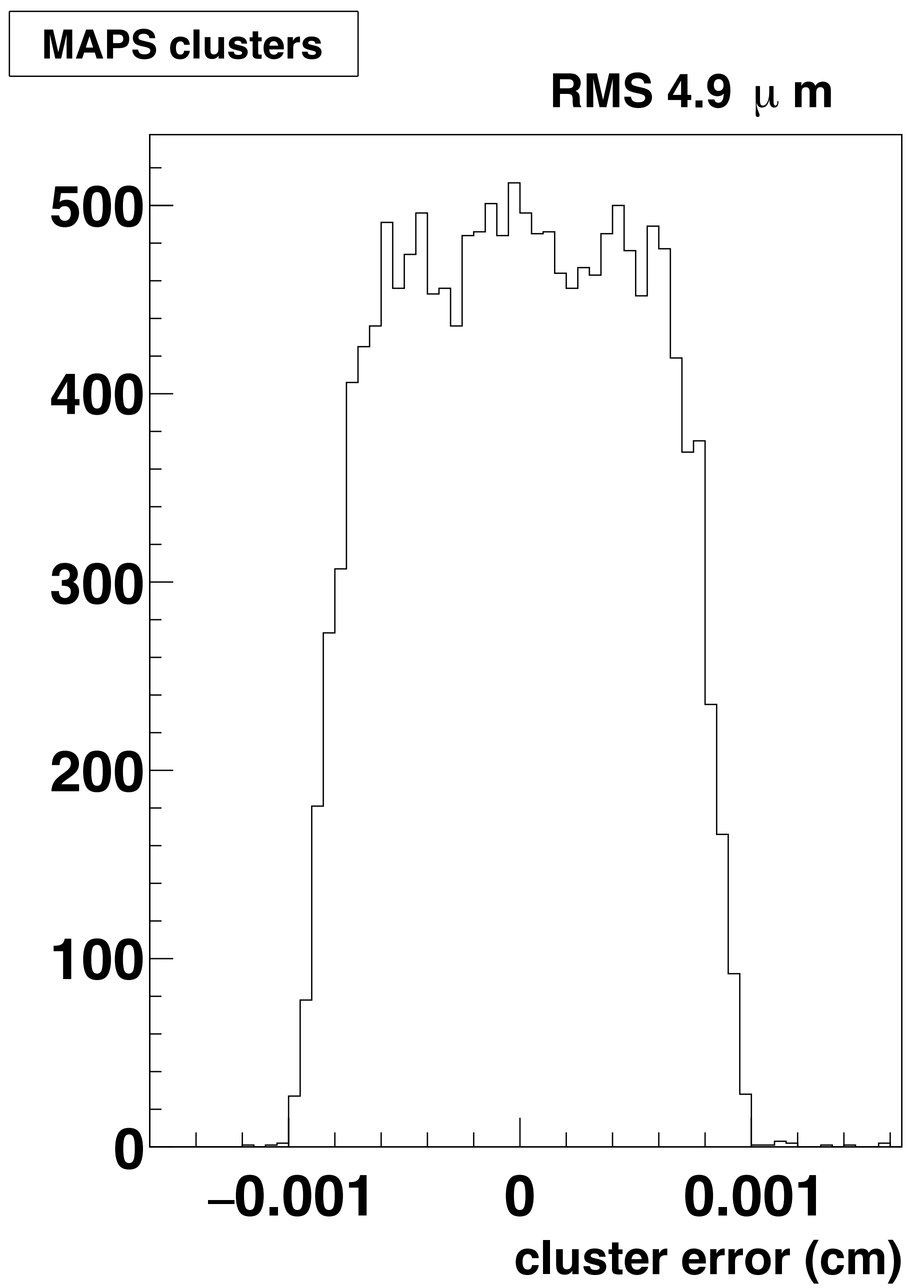
The value of 20 microns/√12 is 5.8 microns. However if we plot the difference in clusterizer position from the truth hit position, we see something different from that.

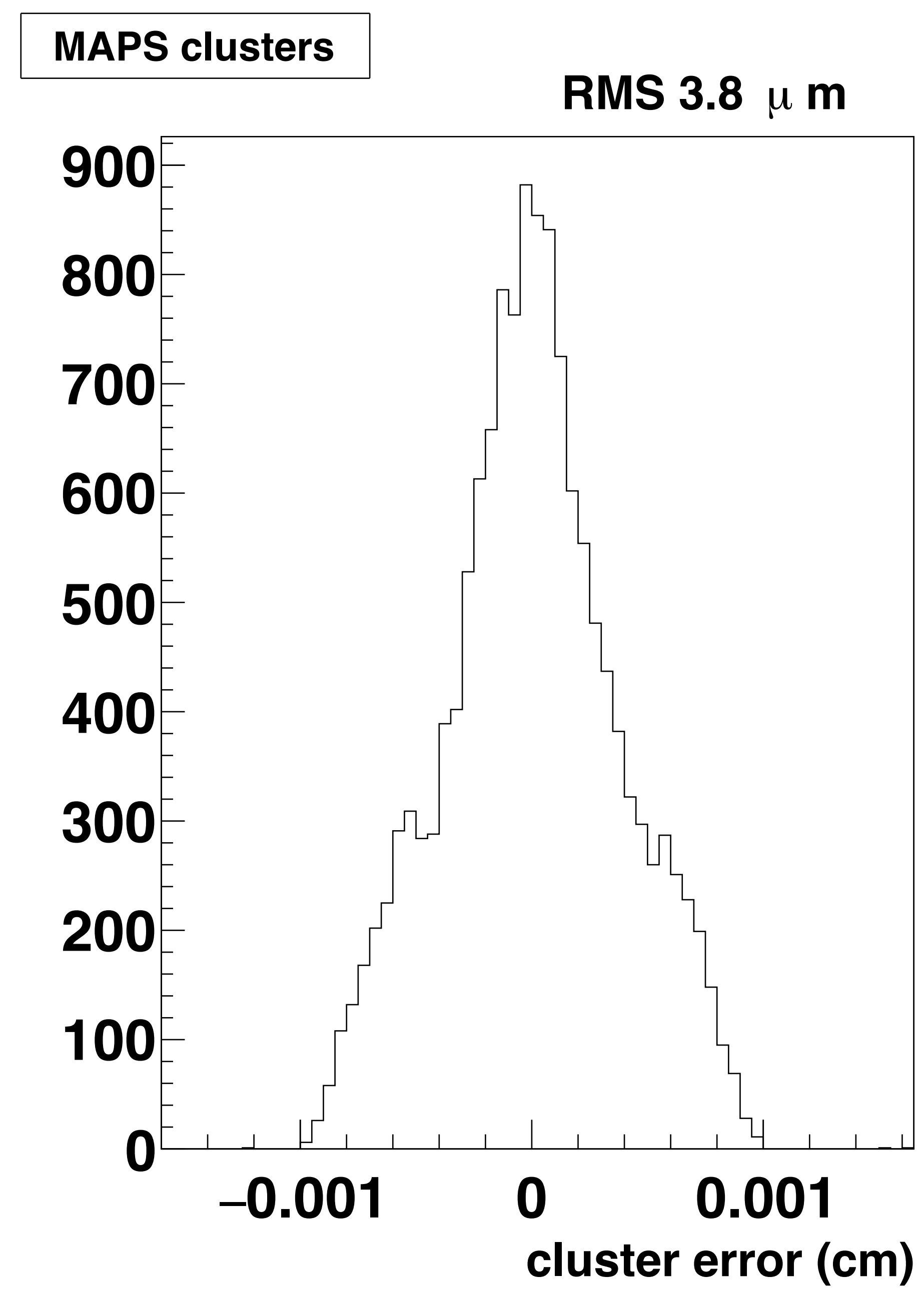




Without charge sharing

The reason for this can be seen if we select clusters with 1 pixel (left) or > 1 pixel (right). having more than 1 hit pixel increases the position resolution quite a bit, and it becomes unrealistically good.





Charge sharing

So, we need charge sharing for a realistic simulation with the ladders. I have implemented a simple charge sharing algorithm for the MAPS pixels.

I found the following thesis to be very helpful:

“Measurements and simulations of MAPS (Monolithic Active Pixel Sensors) response to charged particles - a study towards a vertex detector at the ILC”, by Lucasz Maczewski, Warsaw University (arXiv:10053.3710).

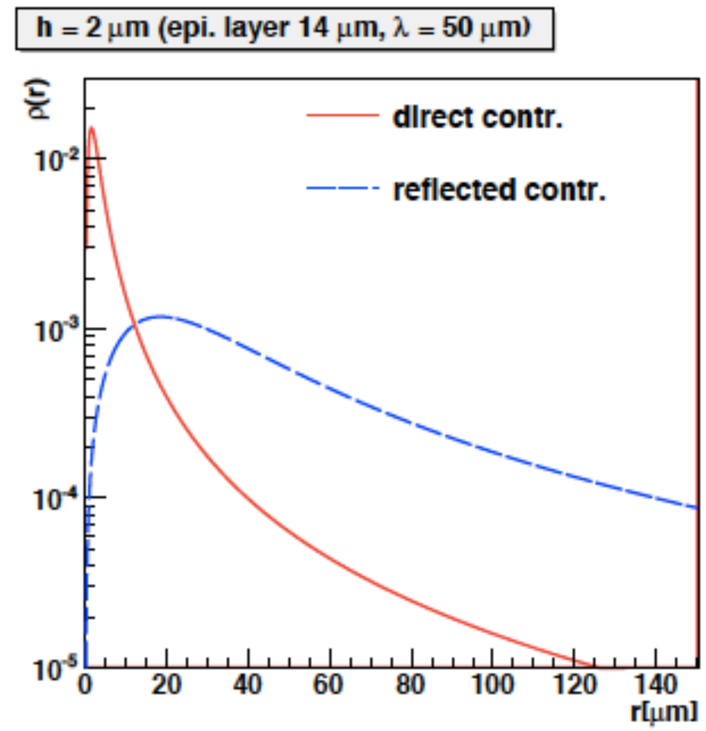
My main takeaway from his model of charge diffusion in the epitaxial layer of a MAPS detector (chapter 7) was how the diffusion width varies with depth in the epitaxial layer (figure 7.3, shown on the next slide).

Diffusion width

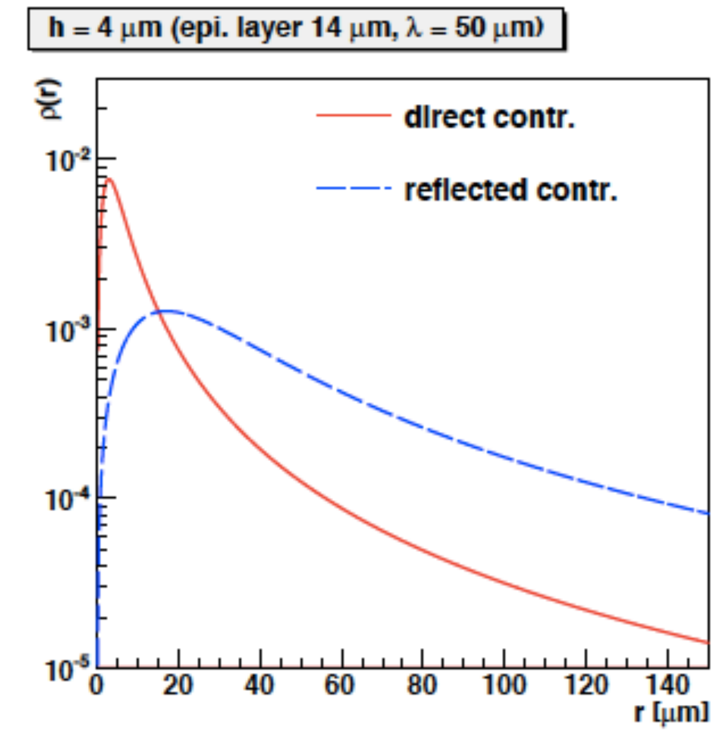
The modeled probability distribution vs diffusion distance for electrons in a 14 µm deep MAPS epitaxial layer.

Shows results for charge formed at four different depths (h) above the collecting diodes.

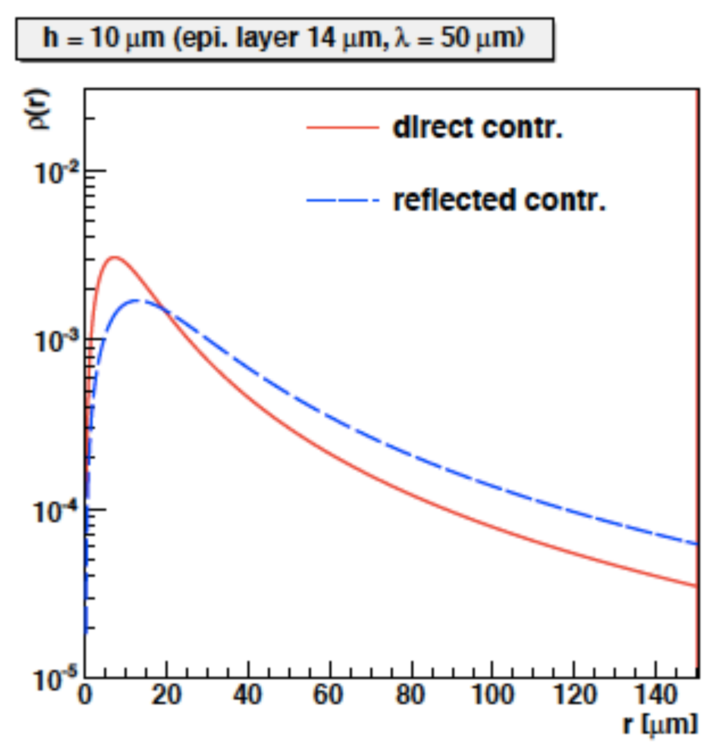
Red curves are for electrons that drift in the direction of the collecting diode. Blue curves are for electrons that drift toward the substrate and are reflected before being collected.



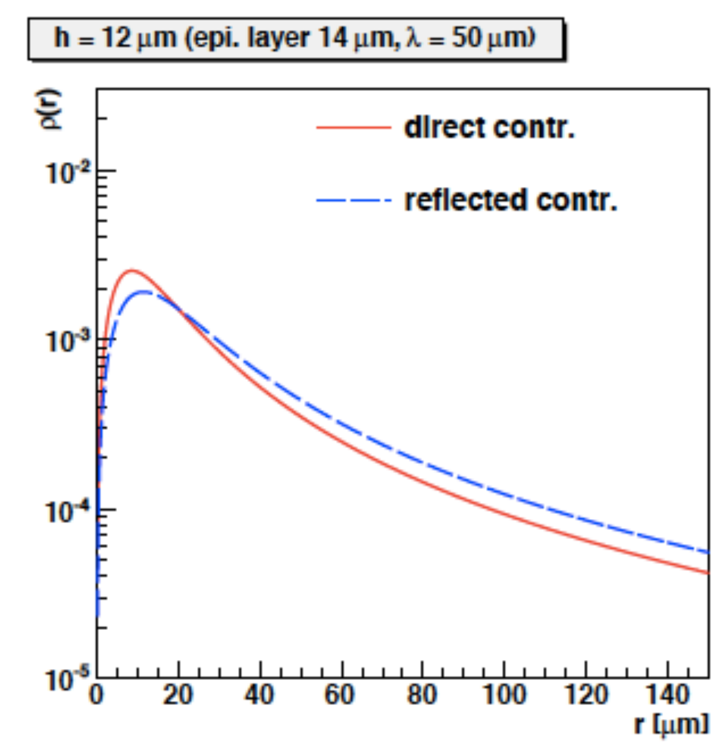
(a) $h = 2\ \mu m$



(b) $h = 4\ \mu m$



(c) $h = 10\ \mu m$



(d) $h = 12\ \mu m$

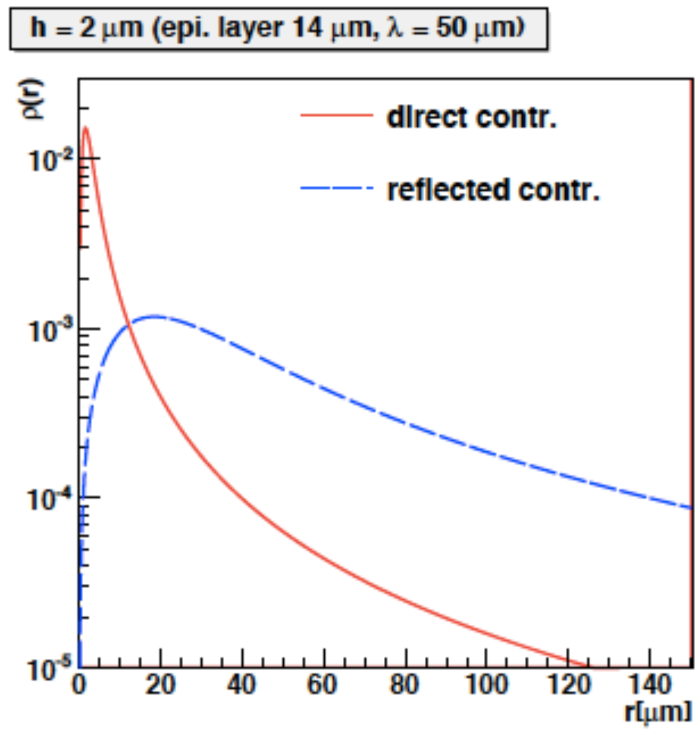
Diffusion width

I did not want to try to implement complicated curves like this.

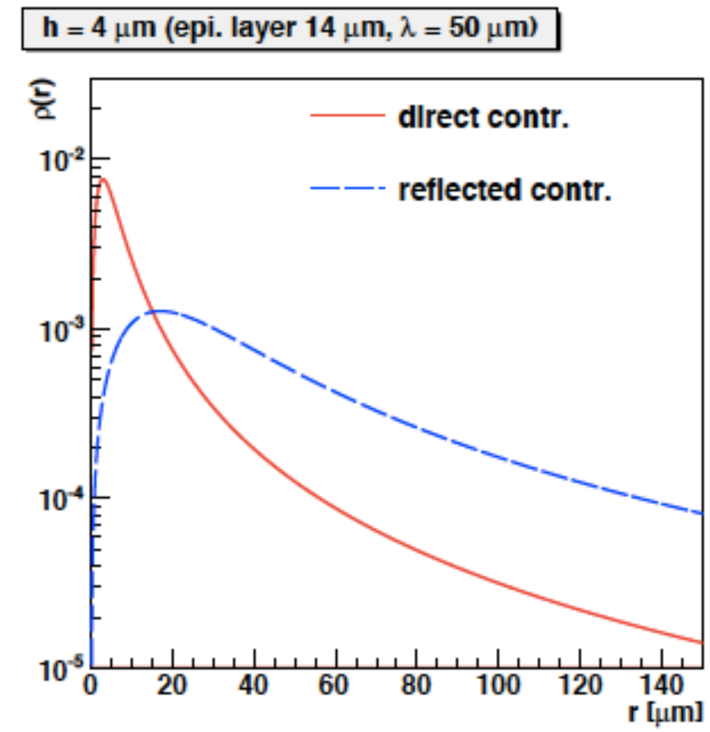
I made the simplification that diffusion produces a circle of electrons of uniform density with:

r = 12 μm (at h = 0)
r = 35 μm (at h = 18 μm).

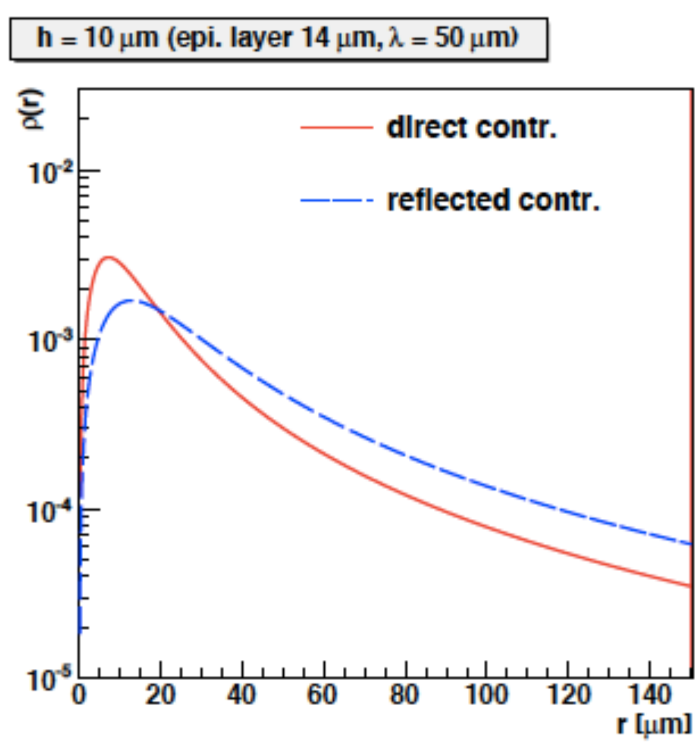
These radii can be varied to see whether they make much difference.



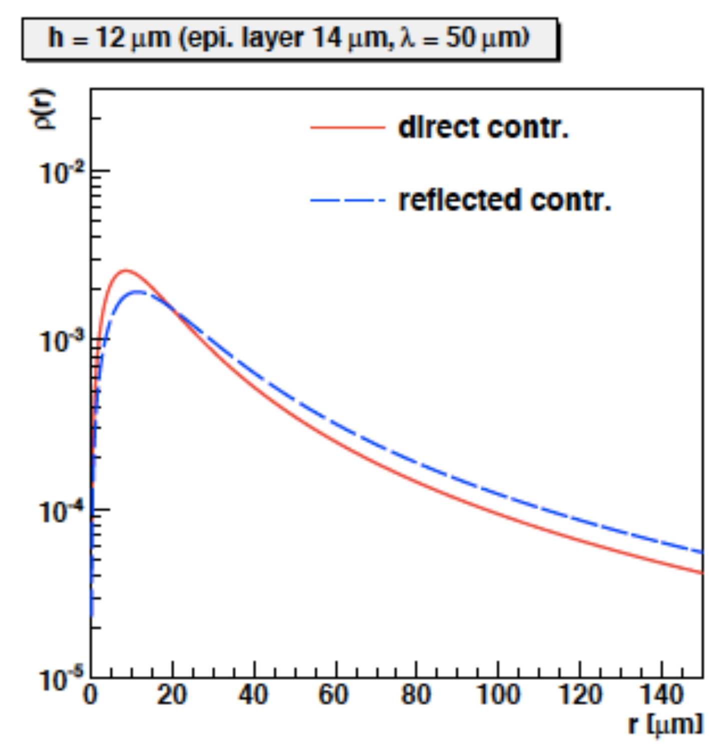
(a) $h = 2\ \mu m$



(b) $h = 4\ \mu m$



(c) $h = 10\ \mu m$



(d) $h = 12\ \mu m$

Diffusion width - cont.

I need to make one more assumption:

The charge collected in a pixel will equal the charge that diffuses into the area of the pixel.

Charge sharing algorithm

I implemented the following simple procedure in PHG4MapsCellreco:

Start with the G4 hit in the MAPS sensor:

- Find the entry location (y_in, z_in) and exit location (y_out, z_out) of the hit in local (sensor) coordinates
- Find the corresponding pixel numbers and get their positions in the (y,z) array of pixels (expressed as (npixel_y, npixel_z))
- Make a list of all pixels between:
  - npixel_y(min) - 2 and npixel_y(max) + 2
  - npixel_z(min) -2 and npixel_z(max) + 2
- Divide the line connecting (y_in, z_in) and (y_out, z_out) into 4 segments (each with 1/4 of the energy) and loop over them:
  - For each segment, calculate the diffusion width (depends on depth)
  - Calculate the overlap area of the diffusion circle with each pixel in the list (uses a general analytic formula)
  - Accumulate the energy deposited in each pixel
- Create a cell (pixel) entry for every cell containing non-zero energy

Code changes

The modified code for PHG4MapsCellReco.(h,C) is committed to “adfrawley/coresoftware" in the “charge_sharing” branch.

https://github.com/adfrawley/coresoftware/tree/charge_sharing/simulation/g4simulation/g4detectors

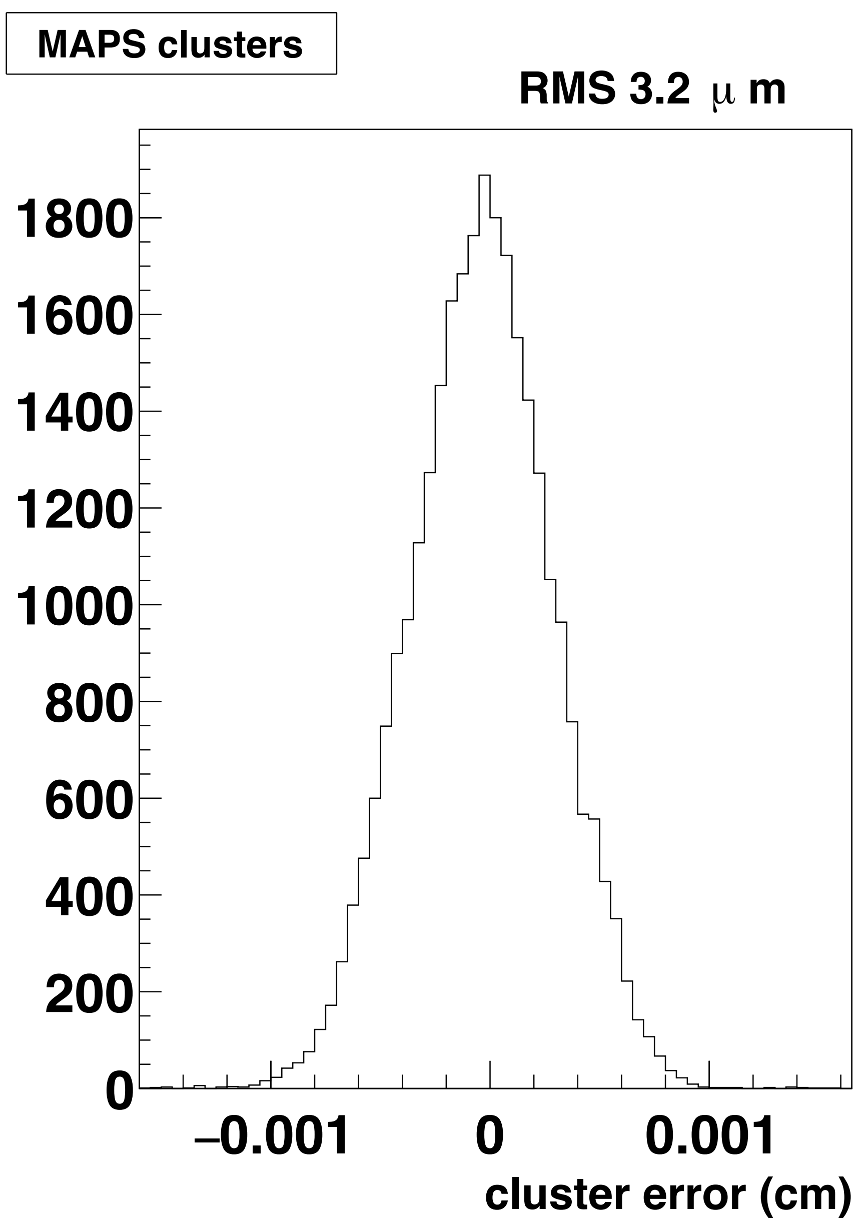
The tests reported here were run using the macro:
G4_Svtx_maps_ladders+intt_ladders+tpc.C
from “adfrawley/macros” in the “QTG_macros” branch.

https://github.com/adfrawley/macros/tree/QTG_macros/macros/g4simulations

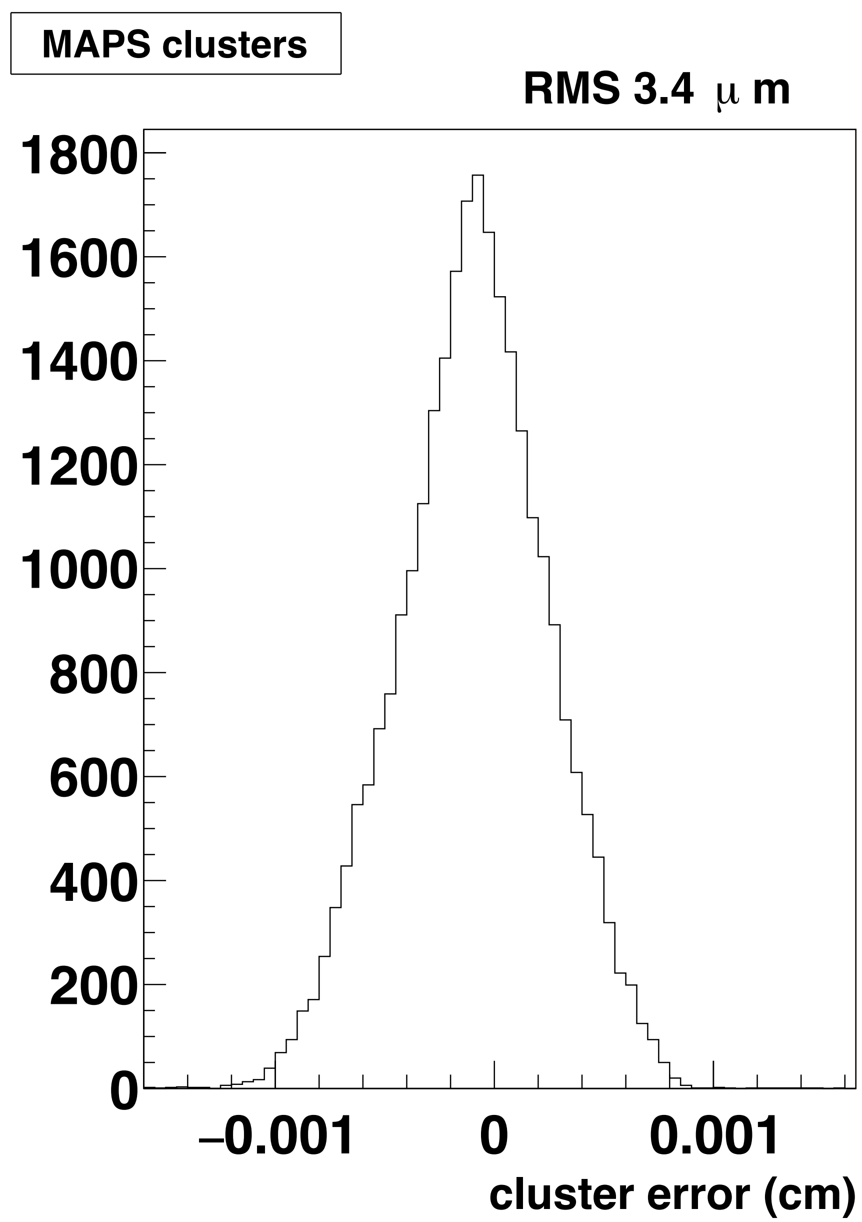
Effect of diffusion constant - cluster errors

From 2-pion events, with 20 μm pixels, I see the following distributions of cluster errors per cluster for three different diffusion constants:

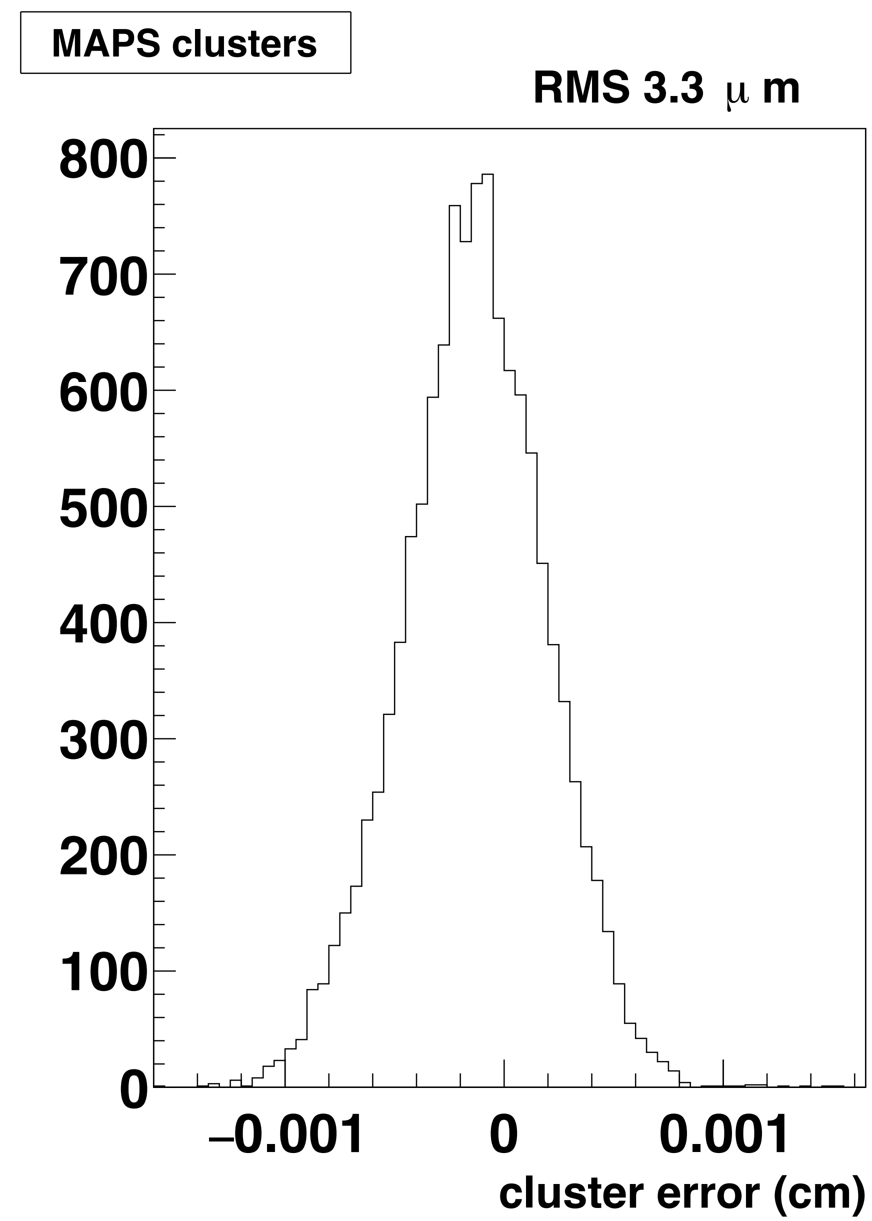
diffusion 25/8



diffusion 35/12



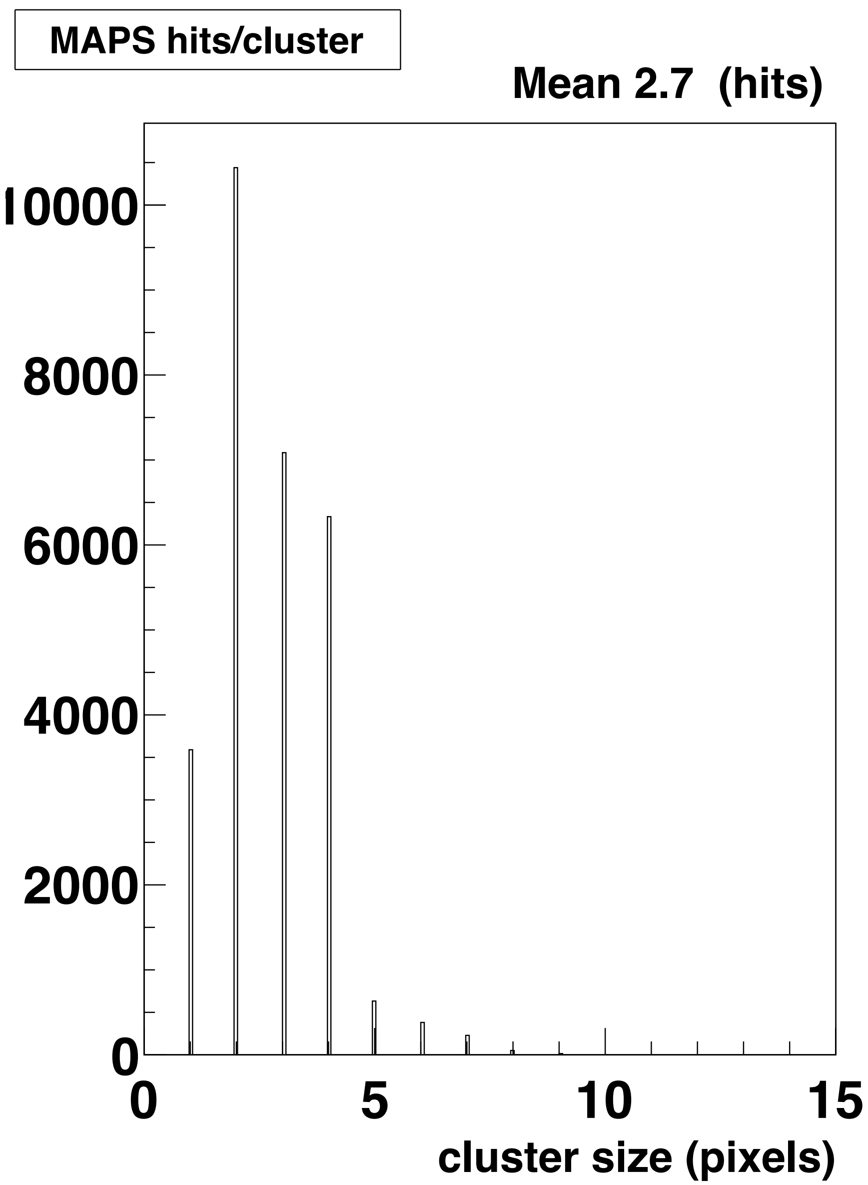
diffusion 45/20



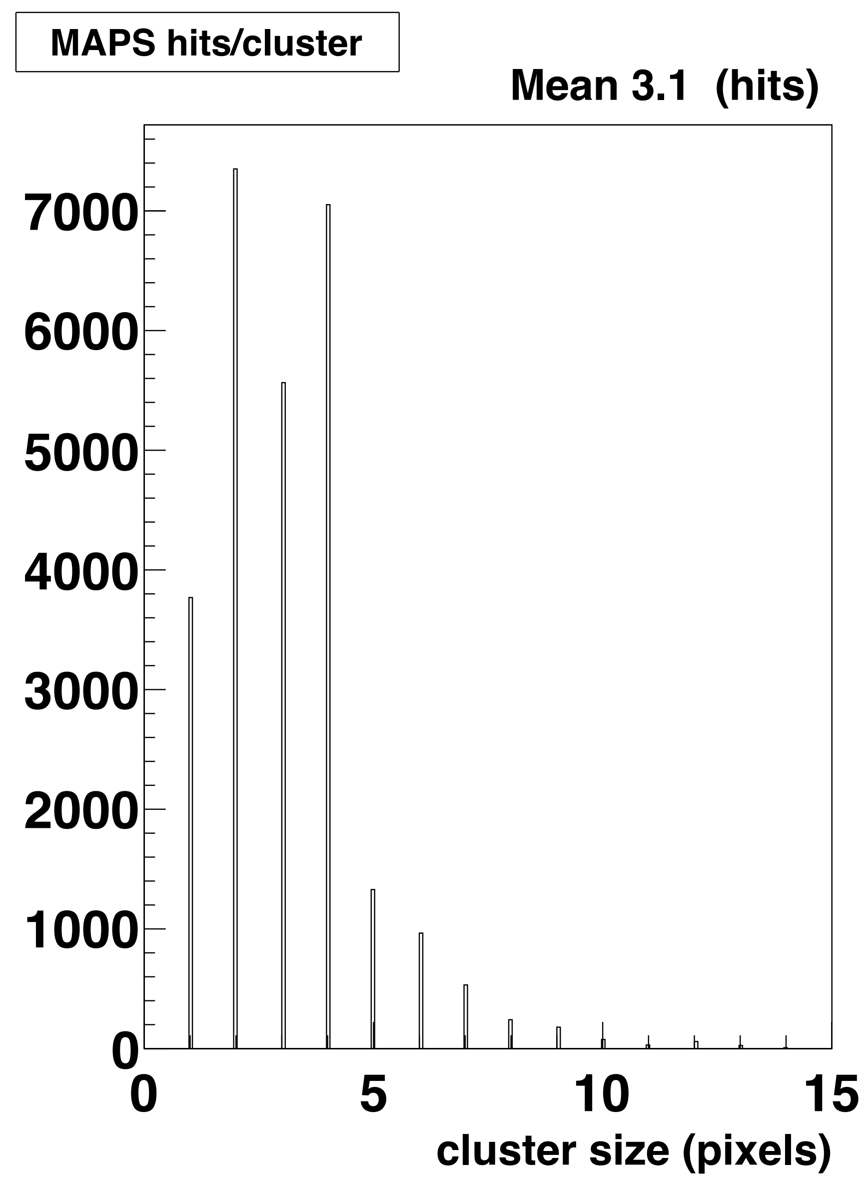
Effect of diffusion constant - pixels/cluster

From 2-pion events, with 20 µm pixels, I see the following distributions of cluster pixel multiplicity per cluster for three different diffusion constants:

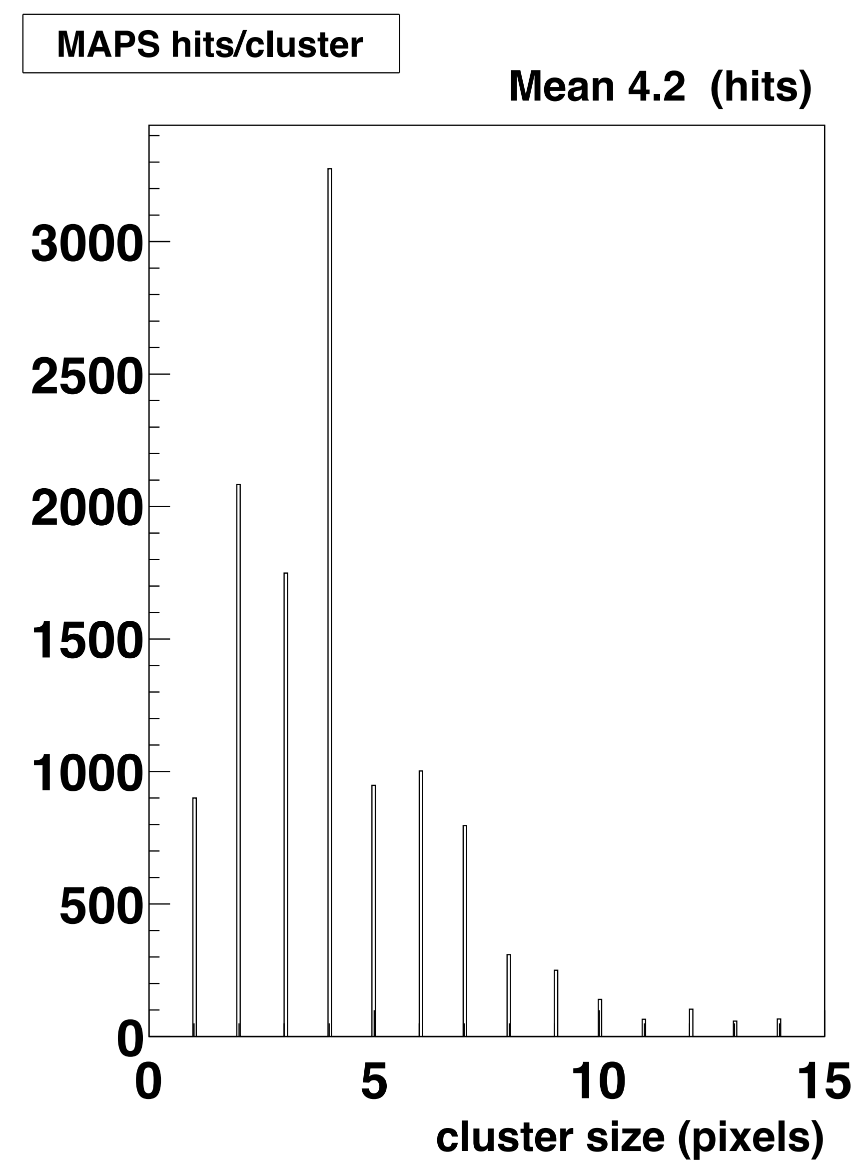
diffusion 25/8



diffusion 35/12



diffusion 45/20



Effect of diffusion constant - summary

Initially I chose the diffusion constant range of 35 µm to 12 µm because that was where the electron probability distributions simulated by Maczewski dropped by ~ 3 from their peak values.

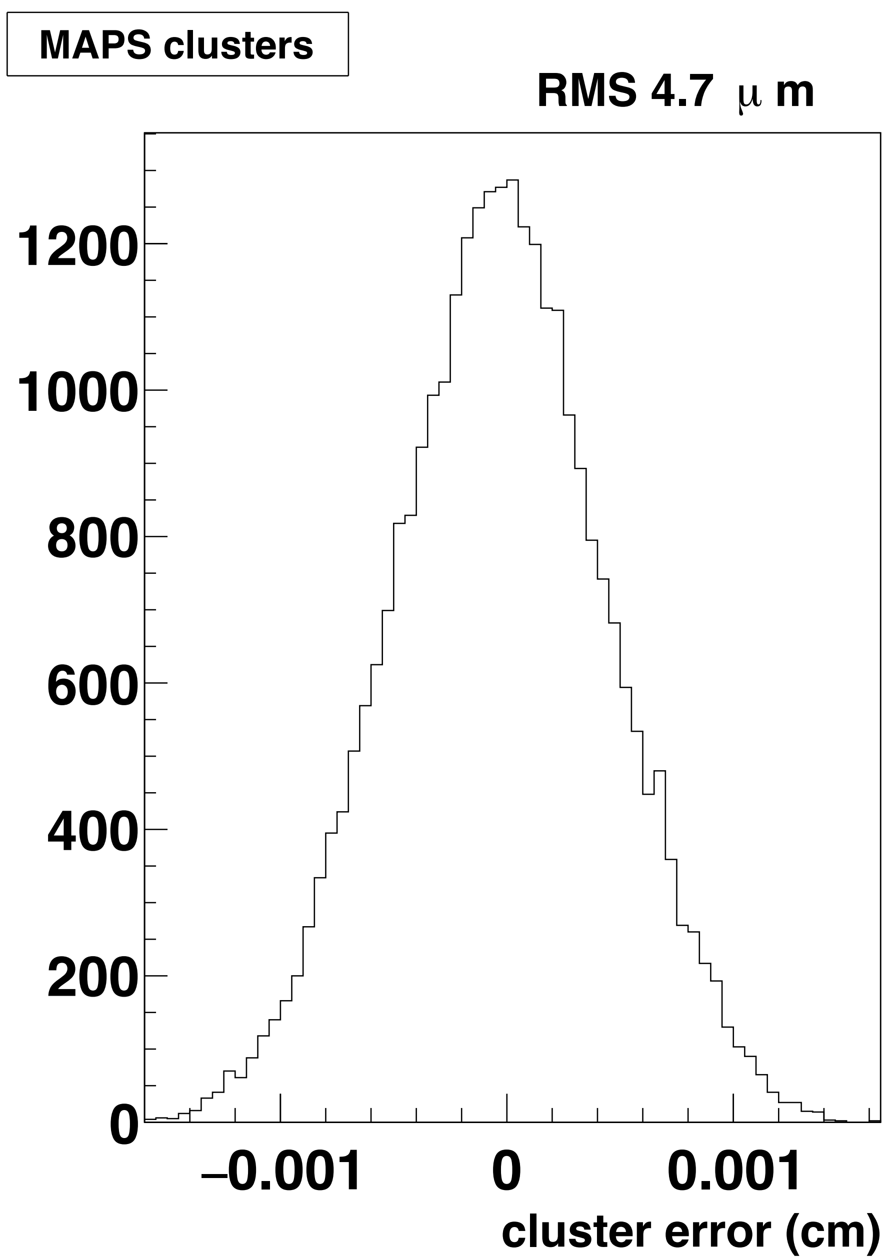
When I repeated the exercise using diffusion constants of (25 µm to 8 µm, seems too low) and (45 µm to 20 µm, seems too high) the cluster resolution values changed by only 1 and 2 µm respectively (< 6%). So I conclude that the cluster resolution is insensitive to the diffusion constants.

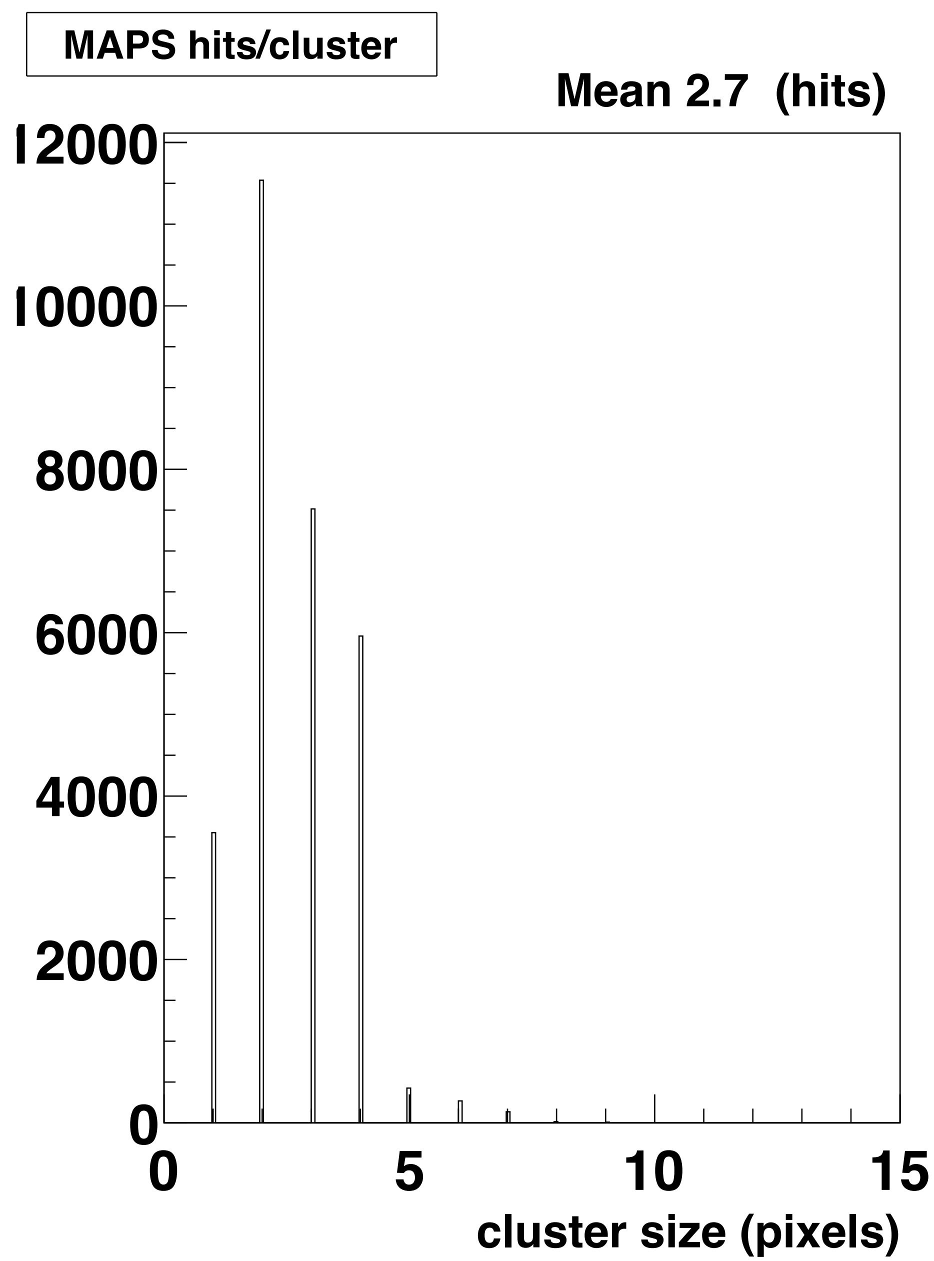
The pixels/cluster were more sensitive to the diffusion constant changes, decreasing by 13% and increasing by 35% respectively. This is not surprising, reflecting just the different areas of the diffusion circle.

I conclude that diffusion constants ranging from 35 µm for charge created furthest from the diode to 12 µm for charge created near the diode should give reasonable results for the cluster resolution and decided to stick with that.

Changing to a realistic pixel size

I changed the pixel size from 20 µm to 30 µm. The pixel size is set in the macro “G4_Svtx_maps_ladders+intt_ladders+tpc.C” during the initialization of “PHG4MapsSubsystem”.





ALICE claims that the resolution should be < 5 µm

Conclusions

The simple charge sharing model that I put into PHG4MapsCellReco seems to produce about the expected cluster resolution for 30 micron pixels, and the result is insensitive to the diffusion parameters used in the model.

Once we have access to the algorithm used by ALICE in their simulations, we can switch to that. But for now I propose that we use this simple charge sharing model for ladder MAPS simulations to overcome the problems outlined in the first few slides. At the same time we should change to using 30 µm square pixels for the maps ladders.